



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE  
United States Patent and Trademark Office  
Address: COMMISSIONER FOR PATENTS  
P.O. Box 1450  
Alexandria, Virginia 22313-1450  
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
-----------------	-------------	----------------------	---------------------	------------------

10/787,005

02/25/2004

Takeshi Ogasawara

JP920030021US1

3932

48233 7590 04/17/2008  
SCULLY, SCOTT, MURPHY & PRESSER, P.C.  
400 GARDEN CITY PLAZA  
SUITE 300  
GARDEN CITY, NY 11530

EXAMINER

BODDEN, EVRAL E

ART UNIT

PAPER NUMBER

2192

MAIL DATE

DELIVERY MODE

04/17/2008

PAPER

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

<b>Office Action Summary</b>	<b>Application No.</b> 10/787,005	<b>Applicant(s)</b> OGASAWARA ET AL.	
	<b>Examiner</b> EVRAL BODDEN	<b>Art Unit</b> 2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

### Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

### Status

- 1) ☒ Responsive to communication(s) filed on 15 January 2008.
- 2a) ☒ This action is **FINAL**.                      2b) ☐ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

### Disposition of Claims

- 4) ☒ Claim(s) 1 - 24 is/are pending in the application.
- 4a) Of the above claim(s) 1, 11-18, 21 and 22 is/are withdrawn from consideration.
- 5) ☒ Claim(s) 5 - 10 is/are allowed.
- 6) ☒ Claim(s) 2-4, 19-20, and 23-24 is/are rejected.
- 7) ☒ Claim(s) 3 is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

### Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

### Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All    b) ☐ Some \*    c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
  2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

### Attachment(s)

- |  |   |
|--|---|
| 1) <input type="checkbox"/> Notice of References Cited (PTO-892)                     | 4) <input type="checkbox"/> Interview Summary (PTO-413)           |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | Paper No(s)/Mail Date. _____                                      |
| 3) <input type="checkbox"/> Information Disclosure Statement(s) (PTO/SB/08)          | 5) <input type="checkbox"/> Notice of Informal Patent Application |
| Paper No(s)/Mail Date _____  | 6) <input type="checkbox"/> Other: _____                          |

### **DETAILED ACTION**

1. This action is in response to the following communication: Amendment to application 10/787,005 filed January, 15, 2008.

**Claims 1, 11 – 18, and 21 - 22** have been cancelled.

**Claim 3** have been amended.

**Claims 2 – 10, 19 – 20, and 23 - 24** remains pending.

**Claims 2, 4, 5, 19, and 23** are independent claims.

#### ***Specification***

2. Prior objections overcome due to correction.

#### ***Claim Objections***

3. Prior objection overcome due to correction.

#### ***Claim Rejections - 35 USC § 112***

4. Prior rejection overcome due to correction.

#### ***Claim Rejections - 35 USC § 101***

5. Prior rejection overcome due to correction.

#### ***Claim Rejections - 35 USC § 102***

6. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

7. **Claims 2, 4, 19 – 20, and 23 - 24** are rejected under 35 U.S.C. 102(b) as being unpatentable over Arnold et al. (hereinafter Arnold) 6,523,168.

In regards to **claim 2**, Arnold teaches:

- an append instruction detection unit for detecting an append instruction (string concatenation) to append a character string to a string variable for storing a character string, in the program (Fig. 5, #94 Does any called method include a string concatenation operation?), (Fig. 4, #74 String Concatenation?, #78 Another Statement in Code?), (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second computer program that require the use of temporary storage), and (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation).
- a store code generation unit for generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit; a store code for storing data of an appendant character string to be appended to the string variable by the append instruction into a buffer, the plurality of append instructions appending the character strings to the same string variable (column 12, lines, 45 – 57, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple

operations in the second computer program that require the use of temporary storage), (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation), and (Fig. 5, #98 Generate Temporary StringBuffer Usage Code, and Fig. 3 #56 Temporary StringBuffer).

- an append code generation unit for generating an append code for appending a plurality of the appendant character strings to the string variable, at a position to be executed before an instruction to refer to the string variable in the program; and the processor further including (column 4, lines 19 – 26, see rather than creating a new mutable string object (as well as an underlying character array object) for each string concatenation operation, an existing mutable string object, allocated at the initialization of a program (or a thread thereof), is used as the temporary storage for each operation. The total number of objects created as a result of multiple string concatenation operations is therefore reduced).
- a reference instruction detection unit for detecting a reference instruction which first refers to the string variable after the character strings have been appended to the string variable by the plurality of append instructions (Fig. 4, #74 String Concatenation?, #78 Another Statement in Code?), and (Fig. 5, #94 Does any called method include a string concatenation operation?).
- the append code generation unit generates the append code at a position to be executed after the store codes and before the reference instruction (Fig. 4, #84 Generate String Concatenation Code), and (Fig. 5, #98, see Generate Temporary StringBuffer Usage Code).

In regards to **claim 4**, Arnold teaches:

- append instruction detection unit for detecting an append instruction (string concatenation) to append a character string to a string variable for storing a character string, in the program; the processor further including (Fig. 5, #94 Does any called method include a string concatenation operation?), (Fig. 4, #74 String Concatenation?, #78 Another Statement in Code?), (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second computer program that require the use of temporary storage), and (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation).
- a store code generation unit for generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit; a store code for storing an address in memory where an appendant character string to be appended to the string variable by the append instruction is stored, into a buffer, the plurality of append instructions appending character strings to the same string variable (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second computer program that require the use of temporary storage), (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation), and (Fig. 5, #98

Generate Temporary StringBuffer Usage Code, and Fig. 3 #56 Temporary StringBuffer).

- an append code generation unit for generating an append code for appending a plurality of the appendant character strings stored in a plurality of the addresses, to the string variable, at a position to be executed before an instruction to refer to the string variable in the program (column 4, lines 19 – 26, see rather than creating a new mutable string object (as well as an underlying character array object) for each string concatenation operation, an existing mutable string object, allocated at the initialization of a program (or a thread thereof), is used as the temporary storage for each operation. The total number of objects created as a result of multiple string concatenation operations is therefore reduced).

In regards to **claim 19**, Arnold teaches:

- detecting an append instruction (string concatenation) to append a character string to a string variable for storing a character string, in the program (Fig. 5, #94 Does any called method include a string concatenation operation?), (Fig. 4, #74 String Concatenation?, #78 Another Statement in Code?), (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second computer program that require the use of temporary storage), and (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation).

- generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit, a store code for storing data of an appendant character string to be appended to the string variable by the append instruction into a buffer, the plurality of append instructions appending the character strings to the same string variable (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second computer program that require the use of temporary storage), (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation), and (Fig. 5, #98 Generate Temporary StringBuffer Usage Code, and Fig. 3 #56 Temporary StringBuffer).
- generating an append code for appending a plurality of the appendant character strings to the string variable, at a position to be executed before an instruction to refer to the string variable in the program, the append code being an optimized set of instructions for a processor to execute (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second computer program that require the use of temporary storage), and (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation).

In regards to **claim 20**, Arnold teaches:

detecting a reference instruction which first refers to the string variable after the



character strings have been appended to the string variable by the plurality of append instructions (Fig. 4, #74 String Concatenation?, #78 Another Statement in Code?), and (Fig. 5, #94 Does any called method include a string concatenation operation?).

the append code generation unit generates the append code at a position to be executed after the store codes and before the reference instruction (Fig. 4, #84 Generate String Concatenation Code), and (Fig. 5, #98, see Generate Temporary StringBuffer Usage Code).

In regards to **claim 23**, Arnold teaches:

- detecting an append instruction (string concatenation) to append a character string to a string variable for storing a character string, in the program (Fig. 5, #94 Does any called method include a string concatenation operation?), (Fig. 4, #74 String Concatenation?, #78 Another Statement in Code?), (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second computer program that require the use of temporary storage), and (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation).
- generating, as a substitute for each of a plurality of the append instructions detected by the append instruction detection unit (column 12, lines 45 – 50, see generating program code for the second computer program that allocates a reusable temporary object for handling multiple operations in the second

computer program that require the use of temporary storage), (column 12, lines 56 - 57, see wherein each of the plurality of operations includes a string concatenation operation), and (Fig. 5, #98 Generate Temporary StringBuffer Usage Code, and Fig. 3 #56 Temporary StringBuffer).

- a store code for storing data of an appendant character string to be appended to the string variable by the append instruction into a buffer, the plurality of append instructions appending the character strings to the same string variable (column 4, lines 19 – 26, see rather than creating a new mutable string object (as well as an underlying character array object) for each string concatenation operation, an existing mutable string object, allocated at the initialization of a program (or a thread thereof), is used as the temporary storage for each operation. The total number of objects created as a result of multiple string concatenation operations is therefore reduced).
- generating an append code for appending a plurality of the appendant character strings to the string variable, at a position to be executed before an instruction to refer to the string variable in the program, the append code being an optimized set of instructions for a processor to execute (Fig. 4, #84 Generate String Concatenation Code), and (Fig. 5, #98, see Generate Temporary StringBuffer Usage Code).

In regards to **claim 24**, Arnold teaches:

detecting a reference instruction which first refers to the string variable after the character strings have been appended to the string variable by the plurality of append

instructions (Fig. 4, #74 String Concatenation?, #78 Another Statement in Code?), and (Fig. 5, #94 Does any called method include a string concatenation operation?). the append code generation unit generates the append code at a position to be executed after the store codes and before the reference instruction (Fig. 4, #84 Generate String Concatenation Code), and (Fig. 5, #98, see Generate Temporary StringBuffer Usage Code).

### ***Response to Arguments***

8. Applicant's arguments with respect to claims 2, 4, 19 – 20, and 23 - 24 have been fully considered but they are unpersuasive. On Page. 7, 2<sup>nd</sup> para. of the response, Applicant contends that "Arnold's patent date is not more than one year from the effective filing date of the present application, and therefore, Arnold is not a proper reference under section 102(b). Specifically, Arnold's patent date is February 18, 2003, and the effective filing date of the present application is February 26, 2003. Thus the rejection of the claims based on the Arnold under section 102(b) is not proper". Examiner respectfully disagrees. 35 U.S.C. 119 reads as follows "(a) An application for patent for an invention filed in this country by any person .... but no patent shall be granted on any application for patent for an invention which had been patented or described in a printed publication in any country more than one year before the date of the actual filing of the application in this country, or which had been in public use or on sale in this country more than one year prior to such filing" (emphasis added). Examiner asserts that the Arnold reference was patented and published on February

Art Unit: 2192

18, 2003, while the present application actual filing of the application in this country is February 25, 2004, which is over a year before the actual filing of the application in this country (emphasis added).

Applicant's arguments on Page. 7, 3<sup>rd</sup> para. fail to comply with 37 CFR 1.111(b) (c) because they amount to a general allegation that the claims define a patentable invention without specifically pointing out how the language of the claims patentably distinguishes them from the references, and they do not clearly point out the patentable novelty which he or she thinks the claims present in view of the state of the art disclosed by the references cited or the objections made.

Furthermore, with respect to claim 2, Page. 7, 3<sup>rd</sup> para., Applicant contends Arnold does not disclose or suggest at least, "generating, as a substitute for each of a plurality of the append instructions ... a store code for storing data of an appendant character string.., into a buffer." "Rather, the cited sections of Arnold disclose using a reusable temporary object for performing the append instructions, emphasis added", and "Arnold appears to execute the append instructions (see Arnold, Col. 10, line 50) themselves". Examiner respectfully disagrees. Examiner notes that the "storing of data of an appendant character string.., into a buffer" has to be kept somewhere, otherwise the compiler device wouldn't be functional. Examiner also notes that Applicant acknowledges that Arnold disclose using a "reusable temporary object", which is by definition "a buffer", and regardless of the "intent", the same objective "storing data of an appendant character sting..., into a buffer" is accomplished.

Arnold also discloses the use of “a buffer” for “storing data of an appendant character string” (Fig. 5, #98 Generate Temporary StringBuffer Usage Code, and Fig. 3 #56 Temporary StringBuffer (emphasis added)). Moreover, Page 7, 3<sup>rd</sup> para., Applicant proclaimed that the intent of Arnold is “using a reusable temporary object for performing the append instructions”, and Arnold “execute the append instructions”, which is different from “storing data into a buffer as a “substitute” for the append instructions”. However, according to Applicant’s abstract, Applicant’s “append code generation unit generates append code for appending a plurality of the append character strings to the string variable”, which is similar to the task that Applicant acknowledges that Arnold accomplished, and which appear to be contradicted with what Applicant’s assertion.

### ***Allowable Subject Matter***

9. **Claim 3** is objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all the limitation of the base claim and any intervening claims.

The following is a statement of reasons for the indication of allowable subject matter: As per claim 3, prior art of record does not teach and/or fairly suggest that “the append instruction detection unit detects, as the append instruction, a combination of: an instruction to convert an immutable string variable in which a process of appending a character string is not allowed, into a mutable string variable in which a process of appending a character string is allowed, an instruction to append the

Art Unit: 2192

appendant character string to the mutable string variable; and an instruction to convert the mutable string variable into the immutable string variable”.

10. **Claims 5 - 10**, prior art of record does not teach and/or fairly suggest a compiler device for optimizing a program which manipulates a character string, the compiler device comprising, “a processor including a mutable-to-immutable conversion instruction detection unit, the processor further including an immutable-to-mutable conversion instruction unit, further including an instruction elimination unit for eliminating the immutable-to-mutable conversion instruction and for causing the mutable string variable to be used as the mutable string variable obtained after the immutable-to-mutable conversion instruction, if an instruction to be executed between the mutable-to-immutable conversion instruction and the immutable-to-mutable conversion instruction does not modify a character string stored in the mutable string variable, and if an instruction to be executed between the immutable-to-mutable conversion instruction and use of the mutable string variable instruction does not modify any of the mutable string variable”, and in such a manner as recited in independent claim 5, thus accordingly dependent claim 6 - 10 are also being allowed.

### ***Conclusion***

11. Applicant's amendment necessitated the new ground(s) of rejection presented in this Office action. Accordingly, **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

***Correspondence Information***

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Evral Bodden whose telephone number is 571 272 3455. The examiner can normally be reached on Monday to Friday, 8:30 to 5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on 571-272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Art Unit: 2192

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Evril Bodden/

/Tuan Q. Dam/  
Supervisory Patent Examiner, Art Unit 2192